INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

Design and Analysis of Algorithms
Department of Mathematics
MA 21007/Assignment-1/Due: 6 Aug. 2014

1.1
Insertion sort can be expressed as a recursive procedure as follows. In order to sort $A[1\,..\,n]$, we recursively sort $A[1\,..\,n-1]$ and then insert $A[n]$ into the sorted array $A[1\,..\,n-1]$. Write a recurrence for the running time of this recursive version of insertion sort.

1.2
Rewrite the MERGE procedure so that it does not use sentinels, instead stopping once either array L or R has had all its elements copied back to A and then copying the remainder of the other array back into A.

1.3
Describe a $\Theta(n \lg n)$-time algorithm that, given a set S of n integers and another integer x, determines whether or not there exist two elements in S whose sum is exactly x.

1.4
Prove that the running time of an algorithm is $\Theta(g(n))$ if and only if its worst-case running time is $O(g(n))$ and its best-case running time is $\Omega(g(n))$.

1.4
The recurrence $T(n) = 7T(n/2)+n^2$ describes the running time of an algorithm A. A competing algorithm A' has a running time of $T'(n) = aT'(n/4) + n^2$. What is the largest integer value for a such that A' is asymptotically faster than A?

1.5
Use the master method to show that the solution to the binary-search recurrence $T(n) = T(n/2) + \Theta(1)$ is $T(n) = \Theta(\lg n)$.

1.6
Can the master method be applied to the recurrence $T(n) = 4T(n/2) + n^2 \lg n$? Why or why not? Give an asymptotic upper bound for this recurrence.

1.7
Consider the regularity condition $af(n/b) \leq cf(n)$ for some constant $c < 1$, which is part of case 3 of the master theorem. Give an example of constants $a \geq 1$ and $b > 1$ and a function $f(n)$ that satisfies all the conditions in case 3 of the master theorem except the regularity condition.

Problem 2-1. Asymptotic Notation

For each of the following statements, decide whether it is **always true**, **never true**, or **sometimes true** for asymptotically nonnegative functions f and g. If it is **always true** or **never true**, explain why. If it is **sometimes true**, give one example for which it is true, and one for which it is false.

(a) $f(n) = O(f(n)^2)$
(b) $f(n) + g(n) = \Theta(\max(f(n), g(n)))$
(c) $f(n) + O(f(n)) = \Theta(f(n))$
(d) $f(n) = \Omega(g(n))$ and $f(n) = o(g(n))$ (note the little-*o*)
(e) $f(n) \neq O(g(n))$ and $g(n) \neq O(f(n))$

Problem 2-2. Recurrences

Give asymptotic upper and lower bounds for $T(n)$ in each of the following recurrences. Assume that $T(n)$ is constant for $n \leq 10$. Make your bounds as tight as possible, and justify your answers.

(a) $T(n) = 2T(n/3) + n \lg n$
(b) $T(n) = 3T(n/5) + \lg^2 n$
(c) $T(n) = T(n/2) + 2^n$
(d) $T(n) = T(\sqrt{n}) + \Theta(\lg \lg n)$
(e) $T(n) = 10T(n/3) + 17n^{1.2}$
(f) $T(n) = 7T(n/2) + n^3$
(g) $T(n) = T(n/2 + \sqrt{n}) + \sqrt{6046}$
(h) $T(n) = T(n-2) + \lg n$
(i) $T(n) = T(n/5) + T(4n/5) + \Theta(n)$
(j) $T(n) = \sqrt{n}\, T(\sqrt{n}) + 100n$

Problem 2-3. Unimodal Search

An array $A[1..n]$ is ***unimodal*** if it consists of an increasing sequence followed by a decreasing sequence, or more precisely, if there is an index $m \in \{1, 2, \ldots, n\}$ such that

- $A[i] < A[i+1]$ for all $1 \leq i < m$, and
- $A[i] > A[i+1]$ for all $m \leq i < n$.

In particular, $A[m]$ is the maximum element, and it is the unique "locally maximum" element surrounded by smaller elements ($A[m-1]$ and $A[m+1]$).

(a) Give an algorithm to compute the maximum element of a unimodal input array $A[1..n]$ in $O(\lg n)$ time. Prove the correctness of your algorithm, and prove the bound on its running time.

A polygon is ***convex*** if all of its internal angles are less than $180°$ (and none of the edges cross each other). Figure 1 shows an example. We represent a convex polygon as an array $V[1..n]$ where each element of the array represents a vertex of the polygon in the form of a coordinate pair (x, y). We are told that $V[1]$ is the vertex with the minimum x coordinate and that the vertices $V[1..n]$ are ordered counterclockwise, as in the figure. You may also assume that the x coordinates of the vertices are all distinct, as are the y coordinates of the vertices.

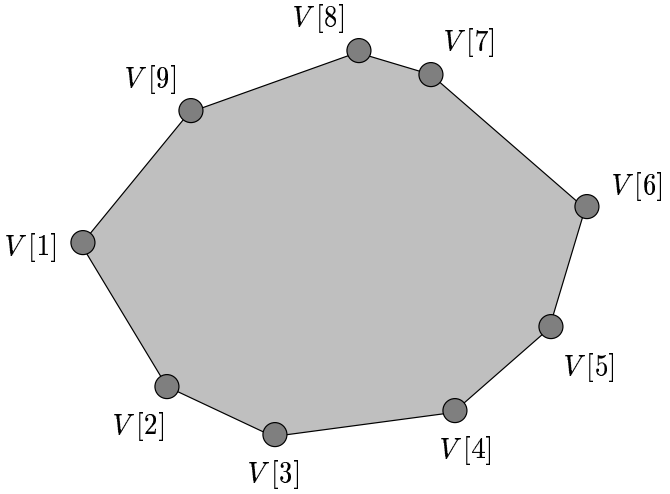

Figure 1: An example of a convex polygon represented by the array $V[1\,..\,9]$. $V[1]$ is the vertex with the minimum x-coordinate, and $V[1\,..\,9]$ are ordered counterclockwise.

(b) Give an algorithm to find the vertex with the maximum x coordinate in $O(\lg n)$ time.

(c) Give an algorithm to find the vertex with the maximum y coordinate in $O(\lg n)$ time.